Approximation de fonction floue à erreur bornée garantie

Vincent Vigneron, Hichem Maaref, Riadh Kallel, Claude Barret

Approximation de fonction floue à erreur bornée garantie

V. Vigneron[1,2], H. Maaref[1], R. Kallel[2], C. Barret[1]

[1]CEMIF-LSC FRE 2494
40 rue du Pelvoux
91020 Evry Cedex, France
vvigne,cbarret,maaref@cemif.univ-evry.fr

[2]MATISSE-SAMOS UMR 8595
90, rue de Tolbiac
75634 Paris cedex 13, France.
vigneron@univ-paris1.fr

Résumé

*Classiquement, un système flou approxime localement une fonction à partir de son graphe au moyen de "patch". Chaque patch définit une règle floue. L'approximation s'améliore en précision au fur et à mesure que le nombre de patchs augmente et que la taille de ces patchs diminue. Dans cet article, nous proposons une autre approche de l'approximation floue : l'estimation des paramètres flous inconnus est réalisée par un calcul ensembliste utilisant les outils de l'analyse par intervalle. Elle fournit l'ensemble des vecteurs de paramètres "*consistant*" avec les données au sens d'une certaine erreur de modélisation qui sera définit dans l'exposé de la méthode. Le but de ce papier est de présenter les premiers résultats obtenus en utilisant cette approche.*

Mots Clef

Ensembles flous, calcul par intervalle, estimation nonlinéaire, inversion ensembliste.

Abstract

Usually, fuzzy systems approximate functions by covering their graphs with fuzzy patches in the input-output state space. Each fuzzy rule defines a fuzzy patch. The approximation increases in accuracy as the fuzzy patches increase in number and decrease in size. In this paper, we propose an other approach for fuzzy approximation in which the estimation of the fuzzy parameters *from experimental data is viewed as a problem of set invertion. It can solved in an approximate but guaranteed way with the tools of interval analysis. Is is, for instance, possible to characterize the set of all parameters vectors that are consistent with the data in the sense that the errors between the data and corresponding model outputs fall within known prior bounds. Any prior knowldge that can be expressed as a series of inequalities to be satisfied by the parameters can be taken into account. The purpose of this paper is to briefly present some results recently obtained in the field of fuzzy approximation.*

Keywords

Fuzzy sets, interval computation, nonlinear estimation, set inversion.

Dans toute la suite, les nombres réels sont en minuscules, les nombres flous en majuscules.

1 Introduction

1.1 Travaux antérieurs

Par définition, un système flou capable d'*apprendre* est *un modèle* dont les paramètres, les entrées et les sorties sont des *nombres flous* qui définit une fonction $f : X \to Y$. Un modèle flou est constitué un ensemble de règles de la forme

$$\text{Si } X \text{ est } A \text{ Alors } Y \text{ est } B.$$

permettant de définir un morphisme $f : X \to Y$ entre deux ensembles flous X et Y. Les règles floues définissent des sous-ensembles appelés *patchs* dans l'espace d'état $X \times Y$. Des patchs larges caractérisent des règles peu précises, de petits patchs des règles plus précises.
Ainsi, sur la figure 1, la règle "Si $X = A_2$ Alors $Y = B_2$" définit un patch dans $X \times Y$ qui couvre le graphe de la fonction f.
En pratique, le calcul sur des nombres flous s'appuie sur l'arithmétique des intervalles, lorsque ces intervalles sont des α-coupe [5]. A titre d'illustration, nous utilisons ici dix niveaux d'α-coupe (*i.e.* $\alpha = 0, 0.1, \ldots, 1$). As shown in Figure 1, the antecedent fuzzy sets $(A_1, A_2, \ldots, A_p)$ of each fuzzy If-Then rule are used as an input vector, and the consequent fuzzy sets $(B_1, B_2, \ldots, B_s)$ are used as the corresponding target vector in the learning. We assume in the following, for sake of simplicity, that the linguistic values are specified by triangular membership functions. It should be noted that a single input (or output) unit handles a fuzzy input (or target) in our model. This is different from other models

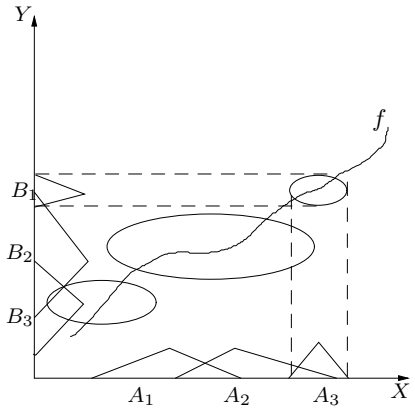


FIG. 1 – Le domaine correspondant à la règle "SI $X = A_2$ ALORS $Y = B_2$" est le produit cartésien flou $A_2 \times B_2$ dans l'espace d'entrée-sortie produit $X \times Y$.

where for example, multiple input (or output) were employed for handling a single fuzzy input (or target) [5]. Since the task of our fuzzified model is to approximately realize the fuzzy IF-THEN rules, the consequent fuzzy sets $(B_1^\alpha, B_2^\alpha, \ldots, B_s^\alpha)$ of each rule are used as a fuzzy target vector when the antecedent fuzzy sets $(A_1^\alpha, A_2^\alpha, \ldots, A_p^\alpha)$ are used as a fuzzy input vector. L'apprentissage d'un modèle flou consiste à minimiser habituellement la différence entre un vecteur flou cible $(B_1^\alpha, B_2^\alpha, \ldots, B_s^\alpha)$ et la sortie floue calculée par le modèle $f(A_1^\alpha, A_2^\alpha, \ldots, A_p^\alpha)$. L'approche d'estimation paramètrique la plus classique consiste à chercher le vecteur de paramètres $\vec{\theta}$ qui minimise le mieux un critère flou J. On peut par exemple chercher les valeurs de $\vec{\theta}$ qui minimise

$$J(\alpha, \vec{\theta}) = \frac{1}{2}\left\{\sum_{i=1}^{s}(B_i^{\alpha+} - f(A_i^{\alpha+}, \vec{\theta}))^2 + \sum_{i=1}^{s}(B_i^{\alpha-} - f(A_i^{\alpha-}, \vec{\theta}))^2\right\}, \quad (1)$$

où les exposants $^+$ et $^-$ désignent respectivement les limites supérieure et inférieure des ensembles α-coupés. La fonction (1) fut proposée par Krishnamraju *et al.* [7] et apprise par Ishibuchi *et al.* [5] au moyen d'un algorithme de descente de gradient.

Habituellement, les algorithmes de gradient sont utilisés pour calculer une estimation des paramètres. Ce sont des algorithmes d'optimisation locaux qui présentent les inconvénients suivants :

1. le choix des valeurs initiales des paramètres sont largement empiriques
2. il n'y a aucune garantie de convergence du critère vers l'optimum global
3. si plusieurs valeurs des paramètres peuvent conduire à la même valeur du coût, il y a un problème d'identifiabilité de ces paramètres et l'algorithme choisira une combinaison possible de ces paramètres (négligeant toutes les autres)
4. en théorie du flou, ce n'est pas tant les valeurs optimales des paramètres qui nous intéressent que de caractériser l'espace des paramètres acceptables.

Un façon de résoudre les 3 premières difficultés est d'avoir recours aux méthodes d'optimisation globales telles que celles proposées par Belforte *et al.* [3]. Pour lever la quatrième difficulté, nous avons choisi de chercher tous les modèles acceptables au sens d'un certain critère au lieu de ne chercher que le meilleur.

Habituellement, une procédure statistique ou neuronale génèrent les règles floues à partir de mesures, quand le système à identifier présente des ambiguïtés, des incertitudes ou qu'il est trop complexe pour qu'on puisse définir des fonctions d'appartenance : les *algorithmes d'apprentissage neuro-flous* ont été proposés par Ichihashi [4], Nomura *et al.* [12], Wang et Mendel [18] indépendamment. De plus, la plupart des méthodes d'extraction des règles floues à partir de données numériques supposent la division *a priori* des variables d'entrées en régions (voir Abe *et al.* [1] and Bauman *et al.* [2]).

Sugeno a décrit dans [14] des systèmes non-linéaires sous la forme d'une collection de sous-systèmes linéaires dont les variables dépendaient d'une mesure d'appartenance du vecteur d'état à différentes régions floues, généralement jointes et définies par des règles SI-ALORS. Son idée principale a été d'utiliser des modèles effectifs autour de certaines régions. Cependant, lLes principaux inconvénients de cette approche multimodèles résident dans la partition des données, qui contrôle les fonctions d'appartenance, et dans la sélection des variables explicatives. Ces tâches sont étroitement reliées les unes aux autres, ce qui complique la modélisation. Le lecteur pourra se reporter également à [14, 15, 19] pour plus de détails sur le raisonnement flou TS du nom de leurs auteurs Takagi et Sugeno. Dickerson and Kosko [6] approxime n'importe quelle fonction avec des règles de forme ellipsoïdale, après "clustering" des données $(x_i, y_i), i = 1, \ldots, n$ autour de centres de classes, n étant le nombre d'échantillons. Les algorithmes de clustering cherchent les règles floues sous-jacentes utilisées par le "processus physique" pour générer les données : autrement dit, le système flou apprend localement la fonction à partir des données d'apprentissage.

L'objet de cet article est de présenter une alternative à ces techniques d'approximation de fonction qui, à

notre connaissance, n'a encore jamais été utilisée en théorie du flou. Notre approche repose sur l'estimation ensembliste (cf. [10] er références incluses). La tâche d'apprentissage est formulée ici comme un problème d'*optimisation* sous contrainte dans lequel une famille de fonctions $\mathcal{F}$ est à découvrir pour relier des ensembles flous A and B par le calcul sur des intervalles. Cette approche regroupe dans un ensemble (à caractériser) tous les vecteurs paramètres solutions. L'algorithme utilisé permet de dimensionner la taille de cet espace solution à partir de la taille des domaines des variables $A \times B$. A la limite, le domaine associé à la règle $A \times B$ est réduit à un point si A et B sont euxmêmes des intervalles de largeur ϵ, avec $\epsilon \approx 0$ (Dans ce cas, la fonction $f \in \mathcal{F}$ peut être estimé en utilisant les techniques d'estmation classiques). Cette approche présente deux avantages : aucune hypothèse sur l'erreur de modélisation n'est requise, et le no statistical assumption on the modeling error is required, and any bounded error can be treated independently from its origin (modeling and/or measurement error).

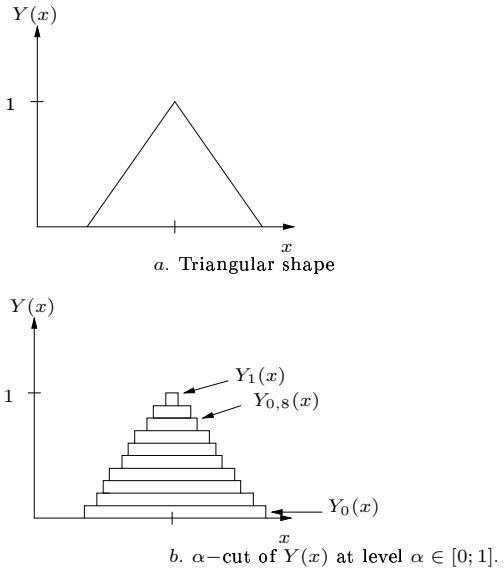
La section 2 présente une formulation générale du problème d'approximation floue. La section 3 rappelle les notions de base de l'analyse par intervalle. La signification pratique des résultats calculés est Our general formulation of the fuzzy approximation problem is mise en évidence dans la section 3.3 à l'aide de quelques exemples numériques.

2 Formulation d'un problème d'approximation flou

Une règle floue approxime une fonction en définissant un patch flou dans l'espace d'état. Une règle floue relie des sous-ensembles flous. La plus simple des règles flous définie une transformation f de $A \times B$ qui associe une sortie floue B avec une entrée floue A. Les règles et les ensembles peuvent avoir n'importe quelle forme : trapézoïdal, *en cloche*, triangulaire, etc. (voir Fig.2.a pour un exemple).

Une règle change quand ses sous-ensembles flous changent. Un ensemble flou exprime le degré d'appartenance à un ensemble : si $\mathcal{Y}$ est un ensemble d'éléments y, un ensemble flou Y dans $\mathcal{Y}$ est défini comme un ensemble ordonné de paires $\{(y, \mu_Y(y)) \mid y \in \mathcal{Y}\}$, où $\mu_Y(y)$ est appelée *fonction d'appartenance* pour l'ensemble flou Y. Si la fonction d'appartenance est à valeur dans $\{0, 1\}$, alors Y se "ramène" à un ensemble ordinaire. Etant donné un ensemble flou Y défini sur $\mathcal{Y}$ et tout $\alpha \in [0, 1]$, l'α-coupe de Y, dénoté Y^α est un ensemble crisp qui consiste dans $\{y \mid \mu_Y(y) \geq \alpha\}$[1].

[1]Similairement, l'$\alpha-$coupe stricte appelée $Y^{\alpha+}$, est définie par $\{y \mid \mu_Y(y) > \alpha\}$.



a. Triangular shape

b. $\alpha-$cut of $Y(x)$ at level $\alpha \in [0; 1]$.

FIG. 2 – Représentation du sous-ensemble flou Y.

Ceci peut s'écrire formellement

$$\mu_Y^\alpha = \alpha \mu_Y(y), y \in \mathcal{Y}. \tag{2}$$

Quand Y est défini sur l'espace des nombre réels $\mathbb{R}$, μ_Y^α sont appelés *fonctions caractéristiques*, et sont définiespour chaque $\alpha \in [0, 1]$ par

$$\mu_Y^\alpha(y) = \begin{cases} \alpha & \forall y \in Y^\alpha \\ 0 & \text{sinon,} \end{cases} \tag{3}$$

avec $Y = \cup_{\alpha \in [0,1]} Y^\alpha$, tel que

$$\left(\cup_{\alpha \in [0,1]} Y^\alpha\right)(y) = \sup_{\alpha \in [0,1]} \mu_Y^\alpha(y), \forall y \in \mathcal{Y}. \tag{4}$$

Chaque ensemble flou est une collection d'ensembles crips imbriqués définissant un tout. In Figure 2.b, a collection of crisp sets which defines the characteristic function μ_Y^α approximates the membership fonction μ_Y at level $\alpha \in [0, 1]$.

Les opérations sur les nombres floues peuvent s'exprimer en terme d'opérations arithmétiques sur des $\alpha-$coupes, qui sont des opérations sur des intervalles bornés de nombres réels.

2.1 Principe

Chaque règle floue SI-ALORS peut être vue comme un couple de nombres flous à associer. Chaque paire d'entrée-sortie floue peut être représenté au niveau α par une paire d'intervalles. L'ensemble de ces couples d'intervalles correspond au couple de nombre flou à associer, *i.e.* chaque paire d'intervalles sera utilisé pendant l'apprentissage du modèle flou comme une entrée-sortie du modèle.

indice de l'exemple	entrée $[x_i]$	sortie $[y_i]$
1	[2 ;6]	[0 ;2]
2	[1 ;5]	[1 ;3]
3	[1 ;5]	[2 ;4]
4	[0 ;4]	[5 ;7]
5	[-1 ;3]	[6 ;8]

TAB. 1 – Exemples d'entrée-sortie (intervalisées) utilisés $([x_i], [y_i])$ dans l'apprentissage du modèle flou.

Ces domaines ont pu être obtenus à partir de mesures approximatives des x_i et des y_i et d'une connaissance des bornes sur les erreurs de mesures commises. Par exemple, supposons qu'un problème flou SI-ALORS est connu de façon approximative à travers les paires d'entrée-sortie $([x_i], [y_i]), i = 1, \ldots, 5$. La Table 1 donnent les supports entrée-sortie qui décrivent intervalles the interval input-output pairs generated from some fuzzy input-output pair. The interval input-output pair are also illustrated on Figure 3. Such interval input-output pairs are used in the learning of the fuzzified model as inputs and targets. The procedure employed consists of estimating the fuzzy parameters associated.
In our learning algorithm, first the value of α and the pattern index are set as $\alpha = 0$ and $p = 1$. So the interval $[2; 6]$ is presented as an input. Next the corresponding interval output from the output unit is calculated by interval arithmetic (see following Section 3). As shown in Table 1, the corresponding target is the interval $[0; 2]$. Then the fuzzy parameters are modified based on the difference e_i between the actual interval output and the target interval. These procedures are iterated for $\alpha = 0, 0.1, \ldots, 1.0$ and $i = 1, \ldots, p$ (here $p = 5$, *i.e.* 5 fuzzy IF-THEN rules).

2.2 Problem formulation

Soit Θ l'ensemble des solutions paramètriques. Let f be some appropriate function. The fuzzy system is assumed to be described by

$$y_i = f(x_i; \vec{\theta}), \quad i = 1, \ldots, p \tag{5}$$

where the observation variable $y_i \in [y_i]$ is related to $x_i \in [x_i]$, a vector of the experiment designed from any linguistic values. It is assumed that f is continuous and differentiable. It is also assumed that domains $[x_i]$ and $[y_i]$ are available. T hese domains can be arbitrarily large, e.g. $[x_i] =]-\infty; \infty[$ if no information is available on a measurement x_i.
In the context of bounded error estimation (e.g. [9] and the references therein), the feasible observation error e_i will be expressed now in terms of a set, *i.e.* $e_i \in \mathbb{E}_i$

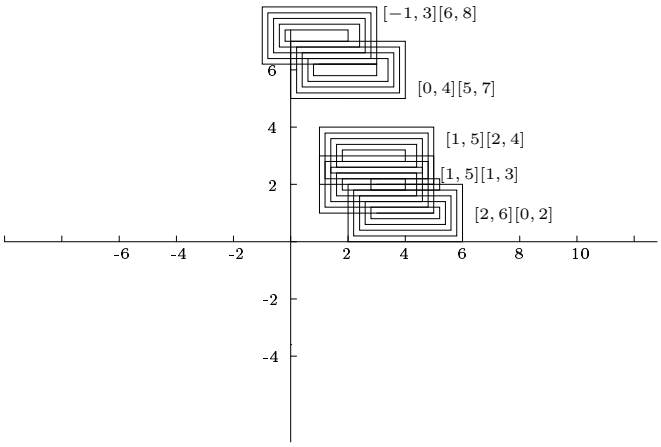


FIG. 3 – α-cut representation of some fuzzy input-output pairs.

where

$$\mathbb{E}_i = \{e_i \in \mathbb{R} \mid e_i^- \leq e_i \leq e_i^+\}. \tag{6}$$

and $i = 1, \ldots, p$ denotes the index pattern. In this equation, e_i^+ and e_i^- are the lower and upper bounds on the observation error, assumed to be known. Estimating the vector $\vec{\theta}$ ($\vec{\theta} \in \Theta \subset \mathbb{R}^m$) of m unknown parameters in this context amounts to looking for the set $\mathbb{S}$ of all admissible values of $\vec{\theta}$ that are consistent with equations (5) and (6). Θ is the prior feasible set for the parameters. $\mathbb{S}$ is thus the intersection of Θ and the set of the solutions for $\vec{\theta}$ of the set of inequalities :

$$y_i - e_i^- \leq f(x_i, \vec{\theta}) \leq y_i - e_i^+, \tag{7}$$

This bounded-error approach leads to set estimates, contrary to usual applications of the maximum-likelihood approach which yields *point estimates*. $\mathbb{S}$ is called the posterior feasible parameter set.
The algorithms used for characterising $\mathbb{S}$ depends on whether the model output f is linear in the parameters. In the first case, it is possible to use an exact description [17]. When the model is nonlinear in the parameters, the problem is much more difficult since $\mathbb{S}$ may be non-convex and even nonconnected. But it is still possible to get a precise description of $\mathbb{S}$ using interval analysis and set-inversion.
Realistic advantages can be found compared to the statistical approach :

1. The error structure is quite simple and similar information usually available in most practical cases, not assuming *any a priori* statistical information about the error.

2. The computation of the parameter domain is conceptually simple and is practically feasible even if the number of data n is not large.
3. The algorithm is *deterministic*.

3 Outils

3.1 Bases de l'inversion ensembliste

Rappelons quelques notions de calcul par intervalle (voir aussi [8]).

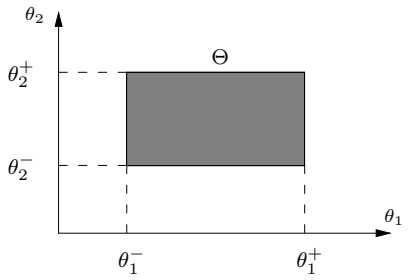


FIG. 4 – Exemple de sous-pavage à deux dimensions.

Un *vecteur d'intervalles* (appelée aussi *pavé*) dans un espace à m-dimensions est le produit cartésien de m intervalles scalaires (voir Figure 4). Il est noté

$$[\vec{\theta}] = [\theta_1^-, \theta_1^+] \times \ldots \times [\theta_m^-, \theta_m^+],$$

où θ_1^- et θ_1^+ désignent les bornes inférieure et supérieure de l'intervalle θ_1, etc. La i-ième composante-intervalle $[\theta_i]$ est la projection de $[\vec{\theta}]$ sur l'axe i. La dimension $w([\vec{\theta}])$ du pavé $[\vec{\theta}]$ est la longueur du plus grand côté.
Cerner l'ensemble des $\vec{\theta} \in \Theta$ satisfaisant (7) revient à trouver l'ensemble solution

$$\mathbb{S} = \{\vec{\theta} \in \Theta | f(x, \vec{\theta}) \in y - \mathbb{E}\}. \quad (8)$$

ou, de façon équivalente,

$$\mathbb{S} = f_\Theta^{-1}(y - \mathbb{E}) = f_\Theta^{-1}(\mathbb{Y}), \quad (9)$$

dans laquelle f_Θ^{-1} est la fonction inverse de f définie sur Θ et $\mathbb{Y}$ est l'ensemble des sorties possibles du modèle. Pour une fonction donnée f, l'*analyse par intervalle* fournit une *fonction d'inclusion* $\mathcal{F}$ qui retourne l'ensemble des pavés de largeur non-nulle contenant l'image de f de n'importe quel pavé $[\vec{\theta}]$ incluse dans le domaine de f. $\mathcal{F}([x])$ est un pavé telle que

$$f([x], [\vec{\theta}]) \subset \mathcal{F}([x], [\vec{\theta}]),$$

et $w([\vec{\theta}]) \to 0 \Rightarrow w(\mathcal{F}([x])) \to 0$. Cette dernière équation est nécessaire pour assurer la convergence.

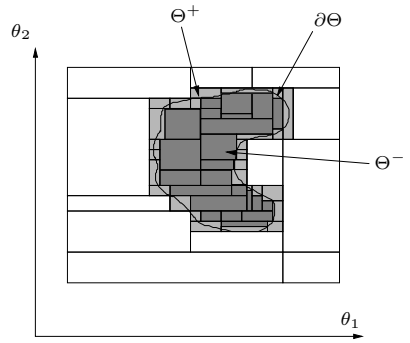


FIG. 5 – Domaines d'inclusion encadrant le domaine des solutions $[\check{\theta}]$.

L'exploration de l'espace des solutions débute par un pavé arbitrairement large $[\vec{\theta}](0)$, qui est ensuite divisé en sous-pavés tant que la largeur de ces pavés est plus grande qu'un facteur de tolérance ϵ_r à fixer. Le calcul par intervalle fournit deux tests pour décider si un pavé donné $[\vec{\theta}]$ est inclu dans $\vec{\theta}$:

$$\mathcal{F}([x], [\vec{\theta}]) \subset [y] \Rightarrow [\vec{\theta}] \in \Theta (i.e.\ [\vec{\theta}] \text{ est acceptable})$$
$$\mathcal{F}([x], [\vec{\theta}]) \cap [y] = \varnothing \Rightarrow [\vec{\theta}] \cap \Theta = \varnothing (i.e.\ [\vec{\theta}] \text{ n'est pas } \textit{acceptable})$$

Dans tous les autres cas, $[\vec{\theta}]$ est *indéterminé*. Si aucune de ces deux conditions n'est satisfaite, l'algorithme calcule par dichotomie deux sous-pavages Θ^- contenant tous les pavés acceptables, et Θ^+ contenant tous les pavés indéterminés (voir Figure 5). Pour tous les Θ, il est toujours possible de trouver deux sous-pavages Θ^- et Θ^+ tel que $\Theta^- \subset \Theta \subset \Theta^+$. Partant de ces sous-pavages, il est facile de borner la portion de Θ contenue dans $[\vec{\theta}](0)$ comme :

$$\Theta^- \subset [\vec{\theta}](0) \cap \Theta \subset \Theta_{out} = \Theta^- \cup \Theta^+. \quad (10)$$

En définissant Θ_{out} comme la réunion des pavés garantie contenir la portion de Θ solution, il est facile de manipuler des ensembles de solutions [9]. Une pile permettant de stocker les pavés étudiés est dépilée à chaque itération.

3.2 Caractérisation de l'ensemble Θ

A l'itération zéro, le pavé à explorer $[\vec{\theta}](0)$ auquel Θ^+ est garanti d'appartenir, est choisi suffisamment large (pour n'oublier aucune solution). Quatre situations peuvent survenir :

1. si $\mathcal{F}([x], [\vec{\theta}](k)$ n'a aucune intersection avec $[y]$, mais n'est pas entièrement dans $[y]$, alors $[\vec{\theta}]$ contient une partie du domaine solution. $[\vec{\theta}]$ est dit *indéterminé*. Si la largeur de ce pavé est plus

grande qu'un paramètre ϵ_r, alors il dit être *bisecté* et le test s'appliquera récursivement aux nouveaux pavés engendrés.

2. si $\mathcal{F}([x],[\vec{\theta}](k))$ a une intersection vide avec $[y]$, alors $[\vec{\theta}]$ n'appartient pas à Θ.
3. si $\mathcal{F}([x],[\vec{\theta}](k))$ est entièrement dans $[y]$, alors $\vec{\theta}$ appartient au sous-pavage solution Θ et s'ajoute aux domaines d'inclusion Θ^- et $\vec{\theta}^+$.
4. Si le pavé considéré est indéterminé, mais de largeur inférieure à ϵ_r, alors il est définitivement adjoint au domaine d'inclusion Θ^+ de Θ.

Au terme de la résolution, aucun pavé n'aura de largeur plus grande que ϵ_r. Θ^+ et Θ_{out} convergeront vers Θ (respectivement par l'extérieur et par l'intérieur) quand $\epsilon_r \to 0$ [13]. L'ensemble Θ^- de tous les pavés solutions retenus peuvent être affiché dans l'espace des paramètres (voir Figure 5).

1.	si $\mathcal{F}([x],[\vec{\theta}](k)) \cap \mathbb{Y} = \varnothing$, retour ;
2.	si $\mathcal{F}([x],[\vec{\theta}](k)) \subset Y$, alors $\{\Theta^- = \Theta^- \cup [\vec{\theta}];\ \Theta^+ = \Theta^+ \cup [\vec{\theta}]\}$, retour ;
3.	si $w([\vec{\theta}](k)) \leq \epsilon_r$, alors ajouter $\{\Theta^+ = \Theta^+ \cup [\vec{\theta}];\ \Theta^+ = \Theta^+ \cup [\vec{\theta}]\}$, retour ;
4.	si la pile n'est pas vide, alors dépiler dans $[\vec{\theta}](k+1)$, incrémenter k de 1 et aller à l'étape 1, sinon stop.

TAB. 2 – Algorithme basée sur une fonction d'inclusion.

La table 2 résume cet agorithme récursif, dans lequel les sous-pavages Θ^- et Θ^+ ont été initialisés vides, et en posant $k = 0$. Jaulin et Walter [9] ont montré -que l'agorithme converge en moins de

$$\left(\frac{w([\vec{\theta}](0))}{\epsilon_r} + 1\right)^n \qquad (11)$$

itérations, le nombre de bisections et le temp de calcul accroissant exponentiellement avec la dimension de $\vec{\theta}$. Le sous-pavage $\partial\Theta \triangleq \Theta^+ \backslash \Theta^-$, constitué de tous les pavés de Θ^+ qui ne sont pas dans Θ^- de largeur inférieure à ϵ_r.
$[\Theta^-;\Theta^+]$ permettent d'encadrer l'enveloppe de l'ensemble solution Θ avec une précision arbitraire, comme illustré sur la figure 6. Sur cette figure, des sous-pavages extérieurs et extérieurs ont été générés pour encadrer l'ensemble des solutions acceptables $\vec{\theta}$. Par rapport aux fonctions caractéristiques floues μ_X^α or μ_Y^α, ces sous-pavages sont les plus précis que les fonctions caractéristiques floues mais plus coûteux en mémoire [13, 17].

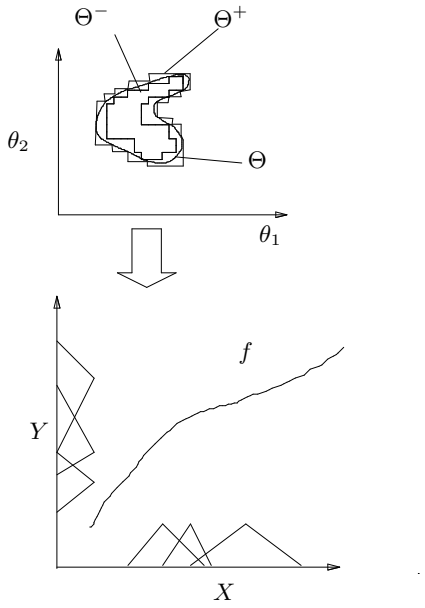


FIG. 6 – Pavage généré pour encadrer l'ensemble Θ des paramètres solutions.

3.3 Propagation de contraintes sur des intervalles α-coupes

Considérons à titre d'exemple, le problème d'estimation à erreurs bornées (voir [9][2]). Il est constitué de deux ensembles flous X, Y et d'un morphisme f de X à Y paramètrisé par $\vec{\theta}$, des fonctions caractéristiques respectives $\mu_X^\alpha(x)$ et $\mu_Y^\alpha(y)$ (pour tout $\alpha \in [0,1]$) et d'un ensemble de contraintes reliant entre elles ces variables. Supposons que 10 couples de mesures $(x_1,y_1),\ldots,(x_{10},y_{10})$ ont été prélevés sur le système considéré. L'ensemble des solutions de ce problème est défini comme l'ensemble $x_i \in [\check{x}_i]$, $i = 1,\ldots,10$ telles que toutes les contraintes soient satisfaites. Les intervalles donnés par la tableau 3.3 sont obtenus en ajoutant l'intervalle $[-2,2]$ à la mesure x_i. Les $[\check{y}_i]$ ont été calculés en ajoutant un bruit-intervalle blanc, centré, de rayon $\rho_i = 0,5w([y_i])$ au vecteur de mesures $\vec{y} = (1,2,3,4,5,6,7,8,9,10)^T$. Les $[\check{x}_i]$,$[\check{y}_i]$ représentent l'$\alpha-$coupe de niveau 0, *i.e.* les supports des ensembles flous X et Y dont les fonctions d'appartenance sont choisies de forme triangulaire (voir Fig.2.a). $[\check{x}_i^\alpha]$,$[\check{y}_i^\alpha]$ sont des intervalles associés à $\mu_X^\alpha(x)$ et $\mu_Y^\alpha(y)$ respectivement. Les boites imbriquées matérialisent l'incertitude associée à chaque paire d'entrée-sortie. La taille de ces boites décroit avec α. Considé-

[2]The extension of the method to multiple-output problems is straightforward.

rons le modèle décrit par les équations

$$y = f(x) = \theta_1 e^{-\theta_2 x} \tag{12}$$

où x, y, θ_1 and θ_2 représente l'entrée, la sortie du système et le vecteur de paramètres[3]. Soit Θ le domaine des solutions.

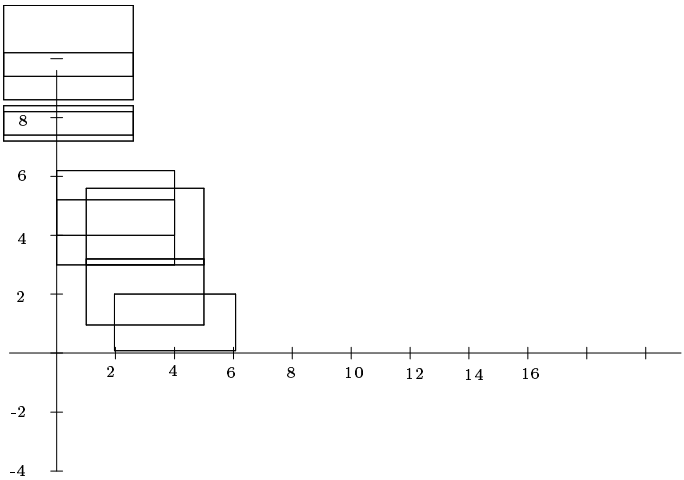


FIG. 7 – Unprecise experimental data.

D'après les Eq. (8) et (2), $\check{\Theta}$ est un ensemble flou défini par une fonction caractéristique μ^{α}_{Θ} telle que

$$\mu^{\alpha}_{\Theta}(\vec{\theta}) = \begin{cases} \alpha & \forall \vec{\theta} \in \check{\Theta}_{\alpha} \\ 0 & \text{otherwise} \end{cases}$$

where $\check{\Theta}_{\alpha}$ is the crisp set that consists of $\{\vec{\theta} \mid \mu_{\Theta}(\vec{\theta}) \geq \alpha\}$.

x_i	$[\check{x}_i]$	y_i	$[\check{y}_i]$
4	[2 ;6]	1	[0.13 ;2.1]
3	[1 ;5]	2	[1.04 ;3.51]
3	[1 ;5]	3	[2.65 ;4.87]
2	[0 ;4]	4	[3.12 ;5.23]
2	[0 ;4]	5	[4.06 ;6.16]
2	[0 ;4]	6	[5.7 ;7.5]
1	[-1 ;3]	7	[6.95 ;8.1]
1	[-1 ;3]	8	[7.57 ;9.38]
1	[-1 ;3]	9	[8.6 ;10.5]
1	[-1 ;3]	10	[9.1 ;11.02]

TAB. 3 – Intervalles de mesure plausibles pour l'exemple considéré.

[3] x and y can be any imprecise variables.

Les domaines pour les paramètres θ_1 et θ_2 sont donnés par

$$[\theta_1] = [-1000; 1000] \text{ and } [\theta_2] = [-1000; 1000],$$

ce qui revient à dire qu'aucune information n'est disponible *a priori* sur les paramètres θ_1 et θ_2. Pour chaque $\alpha-$coupe, les contraintes entre les variables sont données par les équations

$$\begin{cases} y^{\alpha}_1 &= \theta^{\alpha}_1 \exp(-\theta^{\alpha}_2 x^{\alpha}_1) \\ & \vdots \\ y^{\alpha}_{10} &= \theta^{\alpha}_1 \exp(-\theta^{\alpha}_2 x^{\alpha}_{10}) \end{cases} \tag{13}$$

qui nous viennent directement du modèle de l'Eq. 12. La formulation du problème d'estimation peut donc être assimilé à un CSP où les 22 variables sont les x_i, les y_i et les θ_i[4]. L'algorithm donné dans la table 2 peut être utilisé pour caractériser $\check{\Theta}_{\alpha}$, pourvu qu'un test d'inclusion existe

$$\exists \begin{pmatrix} x^{\alpha}_1 \\ \vdots \\ x^{\alpha}_{10} \end{pmatrix} \in \begin{pmatrix} [\check{x}^{\alpha}_1] \\ \vdots \\ [\check{x}^{\alpha}_{10}] \end{pmatrix} \mid \begin{pmatrix} f(\vec{\theta}^{\alpha}, x^{\alpha}_1) \\ \vdots \\ f(\vec{\theta}^{\alpha}, x^{\alpha}_{10}) \end{pmatrix} \in Y^{\alpha} \text{ et } \begin{pmatrix} \theta^{\alpha}_1 \\ \theta^{\alpha}_2 \end{pmatrix} \in \begin{pmatrix} [\check{\theta}^{\alpha}_1] \\ [\check{\theta}^{\alpha}_2] \end{pmatrix} \tag{14}$$

i.e., comme

$$(\vec{\theta}^{\alpha} \in \Theta_{\alpha}) \wedge \phi_1(\vec{\theta}^{\alpha}) \wedge \ldots \wedge \phi_{10}(\vec{\theta}^{\alpha}) \tag{15}$$

où $\phi_i(\vec{\theta}^{\alpha})$ est le test

$$\phi_i(\vec{\theta}^{\alpha}) = (\exists x^{\alpha}_i \in [\check{x}_i] \mid f(x^{\alpha}_i, \vec{\theta}^{\alpha}) \in [\check{y}^{\alpha}_i]). \tag{16}$$

Pour trouver les domaines solutions des paramètres $(\theta^{\alpha}_1, \theta^{\alpha}_2)$ qui sont "*consistants*", au niveau α, avec les équations (13) et les mesures données dans la table 3.3, on définit un algorithme récursif (cf. ci-dessous) qui génère un sous-ensemble de sorties solutions pris dans l'intervalle $[y^{\alpha}_i]$ à partir des entrées $[x^{\alpha}_i]$.
Une valeur pour chaque vecteur variable $\vec{\theta}$ est consistant avec le problème s'il est possible d'instancier $\vec{\theta}$ dans son domaine de façon à ce que les relations (13) soient satisfaites. La contraction sera dîte *optimale* si toutes les valeurs inconsistantes sont otées du domaine solution. La stratégie de contraction des domaines $\Theta_{\alpha}, \alpha \in [0, 1]$ alterne itérativement deux types d'opérations sur les intervalles pour chaque α-coupe :
- une étape de *propagation* : qui calcule l'image $[y^{\alpha}_i]$ de l'intervalle d'entrée $[x^{\alpha}_i]$ par f avec $[\theta^{\alpha}_1], [\theta^{\alpha}_2]$ par le calcul sur les ensembles.

[4] On peut rajouter d'autres contraintes collatérales. Par exemple, si l'on sait que la mesure y_2 est plus grande que la mesure y_3 on rajoutera la contrainte $y_2 \geq y_3$. Le rajout de cette contrainte réduit l'ensemble des solutions.

– une étape de *rétropropagation* : les intervalles $[y_1^\alpha], \ldots, [y_{10}^\alpha]$ obtenus à l'étape précédente sont utilisés pour contracter les ensembles solutions $[\theta_1^\alpha], [\theta_2^\alpha]$.

Le calcul est décomposé en opérations élémentaires, en introduisant volontairement des variables intermédiaires, par exemple

$$\begin{cases} z_2 &= \exp(\theta_1 x_i), \\ z_2 &= \theta_2 z_1, \\ y_i &= z_2. \end{cases}$$

Ainsi, dans l'exemple proposé, le calcul se résume pour chaque valeur de α par l'algorithme suivant :

Algorithme	
entres :	$[x_i^\alpha], i = 1, \ldots, 10,\ [\theta_1^\alpha], [\theta_2^\alpha]$;
initialisation :	$[z_1] := [z_2] :=]-\infty; \infty[$
	rpter
1	$[z_1] := [z_1] \cap \exp([\theta_1] * [x_i^\alpha])$;
2	$[z_2] := [z_2] \cap [\theta_2^\alpha] * [z_1]$;
3	$[\theta_1^\alpha] := [\theta_1^\alpha] \cap [y_i^\alpha]/[z_2]$;
4	$[\theta_1^\alpha] := [\theta_1^\alpha] \cap \log([z_2])/[x_i^\alpha]$;
5	$[z_1] := [z_1] \cap [z_2]/[\theta_1^\alpha]$;
6	$[\theta_2^\alpha] := [\theta_2^\alpha] \cap \log([y_i^\alpha])/[z_1]$;
	tant que la contraction est significative
sortie :	$[\theta_1^\alpha], [\theta_2^\alpha]$;

Les étapes 1-2 traduisent l'étape de *propagation*, les étapes 3-6 l'étape de *rétropropagation*. D'un côté, l'algorithme essaye de partitionner X_α en sous-pavés $[\check{x}_i^\alpha]$ telles que

$$f([\check{x}_i], [\vec{\theta}]) \cap Y_\alpha = \varnothing, \tag{17}$$

pour tous les sous-pavés. Au fur et à mesure de la résolution, tous les sous-pavés X_α retenues (parce que contenant des solutions) sont stockées dans une pile $\mathcal{Q}$. L'algorithme cherche dans chaque sous-pavé un x tel que

$$f([\check{x}], [\vec{\theta}]) \subset Y. \tag{18}$$

Si une telle valeur x est trouvée, cela signifie que $\vec{\theta} \subset \mathbb{S}$. En pratique, un test est introduit à l'étape 6 pour éviter une partition de pavage $[\check{x}_i]$ *ad ifinitum*.

Exemple numérique Le graphe de la figure (3.3.a) est celui de la fonction $y = f(x)$ qui associe les deux ensembles flous donnée par Eq. (12). La figure (3.3.b) donne graphiquement l'allure de la fonction caractéristique consistente avec le système d'équation (13). Cette représentation a lieu dans l'espace paramètrique (θ_1, θ_2). L'algorithme proposé plus haut génère les sous-pavages de la figure 3.3.b en 2,4 secondes sur un `Pentium 800`. Les pavés de couleur grise sont incluses dans l'espace des solutions $\mathbb{S}$, ceux de couleur claire ont une intersection vide avec $\mathbb{S}$. Aucune conclusion n'est donnée pour les pavés de couleur noire. Dans ce contexte, θ_1 et θ_2 sont des *nombres flous* dont les marginales sont les fonctions caractéristiques des paramètres. Le point clé avec cette méthode d'identification est qu'*aucune solution paramètrique n'est oubliée* pendant la résolution.

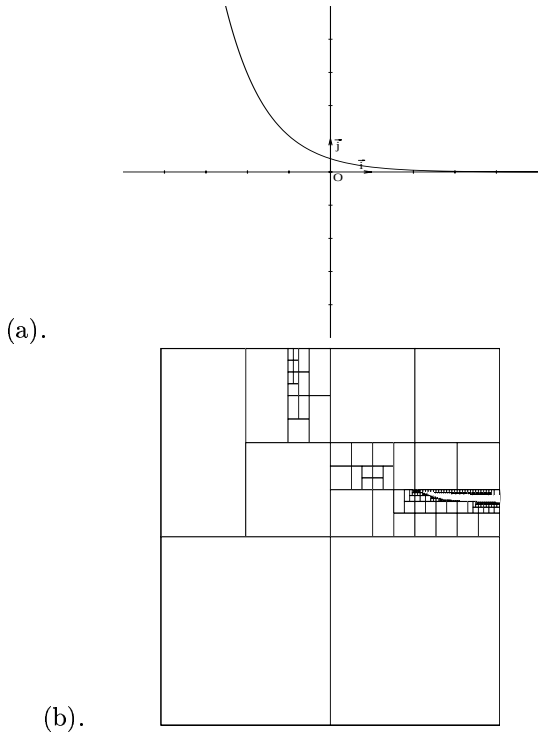


FIG. 8 – (a). Graphe de la fonction f donné par 12. (b). Pavage engendrée par la contraction dans le domaine des paramètres $\check{\Theta}$. La région solution est bornée au niveau $\alpha = 1$ par le pavé $[0.614, 1.78] \times [0.08, 0.095]$.

4 Conclusion

La méthode de résolution ensembliste permet de trouver de façon garantie l'ensemble des paramètres floues d'un problème d'approximation floue, même si le modèle est non-linéaire ou non-identifiable. Les ensembles flous X et Y peuvent comporter un nombre de fonctions d'appartenance arbitrairement grand et fonctionne avec n'importe quelle fonction caractéristique. *A contrario*, les méthodes de gradient utilisent une fonction coût à minimiser consistant en une somme de termes où apparaissent les mêmes paramètres. La multioccurence de ces paramètres est alors inévitable et la fonction d'inclusion de la fonction coût trop péssimiste ce qui se traduit par l'élimination des régions intéressantes du domaine de recherche. Le "prix" à payer

est la complexité exponentielle de l'algorithme en le nombre de paramètres, ce qui restreint son utilisation au problème à peu de dimensions.

Références

[1] Abe, Shigeo and Lan, M-S. (1995). Fuzzy rules extraction directly from numerical data for function approximation, *IEEE Trans. on Syst. Man and Cyb.*, Vol. 25, No 1, 119-129.

[2] Bauman, E., Dorofeyuk, A. and Filev, D. (1990). Fuzzy identification of nonlinear dynamical systems, In *Proc. Int. Conf. on Fuzzy Logic and Neural Nets*, 895-898.

[3] Belforte, G., Bona, B. and Cerone, V. (1990). Parameter estimation algorithms for a set - memebership description of uncertainty, *Automatica*, Vol. 26, No. 5, pp. 887-898.

[4] Ishihashi, H. (1991). Iterative fuzzy modeling and a hierarchical network, Proc. 4th IFSA Congr., Brussels, Vol. Eng., pp. 49-52.

[5] Ishibuchi, H. and Nii, M. (2001), Numerical analysis of the learning of fuzzified neural networks from fuzzy if-then rules, *Fuzzy sets and Systems*, **120**, 281-307.

[6] Dickerson, J.A. and Kosko, B. (1996). Fuzzy function approximation with ellipsoidal rules, *IEEE Trans. on Syst. Man and Cyb.*, Vol. 26, No 4, 542-560.

[7] Krishnamraju, P.V., Buckley, J.J., Reilly, K.D. and Hayashi, Y. (1994). Genetic learning algorithm for fuzzy neural nets. Proceedings of 1994 IEEE International Conference on Fuzzy Systems, pp. 1969-1974.

[8] Jaulin, L. (1994). Solution globale et garantie de problm̀es ensemblistes, application à l'estimation non-linéaire et à la commande robuste, thèse de l'université Paris-Sud.

[9] Jaulin, L. and Walter, E. (1999). Guaranteed bounded-error parameter estimation for nonlinear models with uncertain experimental factors. *Automation*, **35**, 849-856.

[10] Jaulin, L., Kieffer, M., Didrit, O. and Walter, E. (2001). *Applied interval analysis*, Springer, Paris.

[11] Mandani, E.H. (1974). Application of fuzzy algorihtms for control of simple dynamic plant, *Fuzzy sets and Systems*, **28**, 1858-1888.

[12] Nomura, H. Hayashi, I. and Wakami, N. (1991). A self-tuning method of fuzzy control by descent method, a hierarchical network, Proc. 4th IFSA Congr., Brussels, Vol. Eng., pp. 155-158.

[13] Norton, J.P. (1994). Special issue on bounded-error estimation : Issue 1. In *International Symposium Control and Signal Processing*, **8**(1), pp. 1-118.

[14] Sugeno, M. (1988). Structure identification of fuzzy models, *Fuzzy Sets Syst.*, Vol. 28, pp. 15-33.

[15] Takagi, T. and Sugeno, M. (1985). Fuzzy identification of systems and its application to modelling and control, In *IEEE Trans. on Systems, Man and Cybernetics*, Vol. 15, n^{0}1, 116-132.

[16] Tay, T.T. and Tan, S.W. (1997). Fuzzy system as parameter estimator of nonlinear dynamic functions, In *IEEE Trans. on Systems, Man and Cybernetics*, Vol. 27, n^{0}2, 313-322.

[17] Walter, E. (Eds) (1990). Special issue on parameter identification with error bounds. *Mathematics and Computers in Simulation* , **32**(5 et 6), pp. 447-607.

[18] Wang, L.X. and Mendel, J.M. (1992). Backpropagation fuzzy system as nonlinear dynamic system identifiers, Proc. IEEE Int. Conf. on Fuzzy Systems, San Diego, pp. 1409-1416.

[19] Yager, R.R. and Filev, D.P. (1993). Unified structure and parameter identification, *IEEE Trans. on Syst. Man and Cyb.*, Vol. 23, No 4, 1198-1205.